DOCUMENT RESUME

ED 168 552 IR 007 086

AUTHOR Mitchell, William J.
TITLE Teaching Computer Aided Architectural Design at UCLA.
PUB DATE Jul 77
NOTE 6p.; Paper presented at the Computer Aided Design Education International Conference and Exhibition (Middlesbrough, England, July 13-15, 1977); Not available in hard copy due to print quality of the original document

EDRS PRICE MF01 Plus Postage. PC Not Available from EDRS.
DESCRIPTORS *Architectural Education; *Computer Assisted Instruction; Course Objectives; *Curriculum Design; Educational Problems; Instructional Materials; Instructional Staff; Programing Languages; Teaching Methods; Textbooks
IDENTIFIERS *University of California Los Angeles

ABSTRACT

This brief overview includes a rationale for the program and describes course goals and objectives, curriculum content, teaching methods and materials, staffing, and problems of integrating computer aided design with traditional architectural curricula at the School of Architecture and Urban Planning at UCLA. A list of texts for use in teaching computer aided architectural design is appended. (CHV)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

ED168552

U S DEPARTMENT OF HEALTH,
EDUCATION & WELFARE
NATIONAL INSTITUTE OF
EDUCATION

THIS DOCUMENT HAS BEEN REPRODUCED EXACTLY AS RECEIVED FROM THE PERSON OR ORGANIZATION ORIGINATING IT. POINTS OF VIEW OR OPINIONS STATED DO NOT NECESSARILY REPRESENT OFFICIAL NATIONAL INSTITUTE OF EDUCATION POSITION OR POLICY

TEACHING COMPUTER AIDED ARCHITECTURAL DESIGN AT UCLA

"PERMISSSION TO REPRODUCE THIS MATERIAAL HAS BEEN GRANTED BY

William J. Mitchell

TO THE EEDUCATIONAL RESOURCES INFORMATTION CENTER (ERIC) AND USERS OF THE ERIC SYSTEM."

William J. Mitchell

School of Architecture and Urban Planning
University of California, Los Angeles

DP94

Paper presented at the Computer Aided Design Education International Conference and Exhibition, Middlesbrough, England, July 13-15, 1977.

IR007086

TEACHING COMPUTER AIDED ARCHITECTURAL DESIGN AT UCLA

William J. Mitchell
School of Architecture and Urban Planning, University of California, Los Angeles, California, USA.

UCLA is one of the few schools of architecture that teaches computer aided design extensively, and in a systematic and structured way. This paper describes the goals and objectives of the UCLA courses, curriculum content, teaching methods and materials, staffing, and problems of integrating computer aided design with traditional architectural curricula.

1. Why teach computer aided design to architecture students?

Since the architectural profession generally makes very little use of computer aided design techniques in practice, it is fair to ask what relevance computer aided design education has for architecture students. There are two answers.

The first is that the lack of widespread application of computer aided architectural design techniques in practice is a temporary condition which will have changed by the time the present generation of students graduates in the 1980's. Two major barriers have stood in the way of implementation of computer aided design systems in architectural offices: the high cost of appropriate computer hardware, and the low level of development of geometric description software. Both these problems are now very close to solution.

The most dramatic reductions in hardware cost have been due to the emergence first of minicomputers, then of LSI microprocessors. The introduction of storage tube terminals sharply reduced the cost of interactive graphics. Low cost raster-scan graphics, plasma panel systems, and new refreshed tube system architectures are further important developments in the area of low cost graphics. Amortized over a reasonable period, a minicomputer-based graphics system suitable for most architectural applications now need cost an office no more per year than a single architectural draftsman. Further substantial cost reductions can be expected.

To achieve the full potential benefits of computer aided design in architecture, it is necessary to have a data base facility capable of maintaining a full, three-dimensional geometric description of a building. This type of facility is difficult to develop because complete building description requires a large quantity of data (of the order five to fifty million bytes) and complex data structure (variable-format self-describing records, multiple indexing, procedural modeling of geometry). Furthermore, real-time interactive graphic manipulation of the design is needed. However, Applied Research of Cambridge's OXSYS system (1) the Evans and Sutherland Design System (2), and the BDS (3) and GLIDE (4) systems developed by Eastman at Carnegie-Mellon University, have demonstrated that this type of facility can be implemented successfully on a minicomputer. A recent feasibility study (5) concluded that a large, general system capable of efficiently handling any building

type, geometry, or method of construction is now possible.

These developments will soon remove the remaining technical and economic difficulties associated with practical computer aided architectural design. Acceptance of computer aided design techniques still may not come easily, since the architectural profession is notoriously conservative. But architects who do not take use of them will soon find it very difficult to compete in time, cost, and quality of work with those who do.

The second reason for teaching computer aided design to architecture students has nothing directly to do with current architectural practice. The constructs and vocabulary of information processing provide a useful, rigorous framework within which to research and discuss fundamental questions of design theory and method. Herbert Simon's writings on the "sciences of the artificial" (6) exemplify the insights which this viewpoint can give.

I would argue that this second reason is of the utmost importance within a university. Our primary obligation is not so much to provide direct training for practice as to develop an understanding in students of basic concepts which provide a firm foundation for continued learning throughout a long professional career. Within a framework of information processing constructs, many architectural concepts which traditionally have been taught only by example and apprenticeship can now be concisely and rigorously expressed at a theoretical level.

2. Curriculum objectives

The basic curriculum objective should be to develop an understanding of fundamental concepts, i.e. modeling, languages, algorithms, and data structures, and how these relate to design problems. This requires both formal instruction by means of lectures, seminars and readings, and as much practical computing experience as possible. Just as architecture is taught conventionally by means of studio problems, so computer aided design must be taught largely by getting students to work non-trivial problems using a computer.

Involvement of students at the programming language level is absolutely essential. Anything less does not give adequate theoretical insight into the concepts of algorithms and data structures, and in fact can be positively damaging. I regard the widespread practice of teaching survey courses of packaged application programs as particularly pernicious. Learning the input and output formats of an arbitrary collection of programs, most of which are likely to be obsolete, is quite useless. Students need to know how to approach real problems creatively and critically, not how to enter predefined data to generate answers to predefined puzzles.

It is sometimes objected that architecture students are not "mathematical" and therefore cannot or do not want to learn to program. But I suggest that any difficulty experienced in teaching programming is more likely to be a result of inappropriate teaching methods than of any inherent lack of ability in the students. As evidence, it is worth noting that normal primary school children can be taught to write quite sophisticated programs in LOGO, and that electronic calculators based upon the supposedly difficult concepts of an operand stack and reverse Polish notation have been a commercial success. In any case, our experience at UCLA over many years has clearly demonstrated that the objection is not valid.

3. Curriculum content

The diagram illustrates the structure of computer aided design courses currently available to architecture students at UCLA. Each course lasts ten weeks, with two meetings per week. The students are either in a first professional degree program (which is taught at the graduate level) or in a second professional M. Arch. degree program. In keeping with the general philosophy of the UCLA curriculum, none of these courses is required. Students have the opportunity to take some, all or none.

Most architecture students take one or two of the introductory courses, and only those who wish to specialize take the full sequence. The classes usually also contain a few students from urban planning, fine arts, operations research, and computer science.

The basic introductory courses are a programming course (in FORTRAN) and a seminar in design theory and method. The programming course is intensive. The students are expected to write and run a simple program by the day after the first class, to complete weekly programming exercises of increasing difficulty, and to emerge at the end of ten weeks as reasonably competent programmers. Most do. The emphasis throughout is upon learning how to model problems and formalize the steps required for their solution, rather than upon detailed examination of the particular features of FORTRAN.

The design theory and method course focuses upon the basic theory of problem solving from an information processing point of view. It consists of a mixture of lectures, weekly assigned readings, and seminar discussions of the lectures and readings. It aims to provide a general perspective which enables a student to compare, contrast and evaluate various different traditional and computer-based approaches to design.

At the next level are courses in mathematical modeling computer graphics and information systems. The modeling course has no prerequisites, but the programming and design theory courses are recommended preparation. It begins with fundamentals, i.e. sets, transformations, matrices, and groups, moves on to graph theory, then selected topics in combinatorics and mathematical programming. Completion of weekly problem sets is required. The intention is to introduce, at a very basic level, the mathematical concepts most commonly employed in modeling architectural problems. One might expect that students who have experienced modern mathematical curricula in primary school and high school would not need much of this course. But we find that most students either are unfamiliar with the material, have forgotten it, or have been so badly taught that it is better to start again at the beginning.

The graphics course requires a programming background, and must be taken simultaneously with the modeling course. It covers graphics devices, two and three-dimensional transformations, and selected topics in advanced graphics techniques. The work on transformation matrices is closely coordinated with the mathematical models course, which provides the necessary background. Practical exercises begin with writing subroutines to draw simple graphic figures on a storage tube, then transformation routines. As a final project, students must write a non-trivial interactive program for the storage tube, using cross-hair or tablet input.

The information systems course also requires a programming background. It covers data structures and data base management technology, with particular emphasis upon architectural applications. Students in the course jointly write a simple data base management system for an architectural application.

Together, the graphics and information systems courses provide preparation for the computer aided design course. This is a course in which students work jointly upon design and implementation of some kind of architectural computer aided design system. In parallel with this are visiting lectures from workers in the field, seminars on selected theoretical topics, and case studies of existing systems.

Students completing the full sequence of courses typically go on to supervised individual study leading to a thesis. Thesis topics often arise out of problems initiated in the graphics, information systems, or computer aided design courses.

4. Teaching methods and materials

It can be seen from the curriculum description that there is a strong emphasis upon learning by working on problems. To support this, the School of Architecture and

Urban Planning has a storage tube graphics system, equipped with a tablet input device and electrostatic printer/plotter, located in the design studio. It is continuously available for student use, and a full time consultant (who is a member of the school staff) is in attendance.

A vital feature of the approach is that all the courses which have been described are taught within the School of Architecture and Urban Planning by faculty members of the School. The reason for this is that we have found it to be essential to introduce and illustrate all concepts by means of specifically architectural examples. If this is not done, students find it almost impossible to relate computational concepts to their architectural work. Courses must be specifically tailored for architecture students and taught by people with combined backgrounds in architecture and computing. Service courses from other departments in principle are not adequate. They might superficially appear to cover the same topics, but they cannot be sufficiently sensitive to the architectural implications.

FORTRAN is the basic teaching language used throughout. This is for reasons of history and convenience rather than principle. In fact, FORTRAN is a highly undesirable language for building complex data structures, for graphics, and for interactive work (all vital for architectural applications). Furthermore, its cumbersome syntax obscures rather than elucidates the principles underlying an algorithm. It would be much better to use a more up-to-date, higher level language like ALGOL 68, PASCAL, EULER or PL/1. Better yet would be to use a language specifically intended for geometric modeling applications, like GLIDE (7) or the E&S Design System language (8). A change in this direction can be expected at some point in the future.

A major difficulty in the past has been the lack of suitable text books. It was necessary to rely heavily upon mimeographed notes, and collections of readings from widely scattered sources. However, a number of useful texts have been published in the last few years, and the situation now is much better. A short list is given in an appendix to this paper.

5. Staffing

Finding teaching staff qualified to teach the types of courses that have been described here can be a difficult problem. As yet, there are few architects with sufficient computing background, and conversely, few computer scientists or technologists with sufficient architectural background.

To overcome this difficulty, we have found that joint-teaching of a course by two people, each providing different aspects of the required background, can be very successful. For example, the mathematical modeling course was at first jointly taught by an architect and an applied mathematician. After several years experience, each can now teach it independently.

6. Integration with existing curriculum structures

Integration of computer aided design education with existing architectural curricula is another difficult issue. On the one hand, architectural curricula tend to be overcrowded already. On the other hand, it is useless to approach computer aided architectural design in a superficial way. The only solution is for this subject to replace some of the traditional content of an architectural curriculum. This is conceptually as well as practically appropriate, since the development of computer aided architectural design makes some of the traditional content obsolete. It cannot be expected, however, that this replacement will take place quickly or easily in most schools.

7. Conclusion

The basis of the approach to computer aided architectural design education at UCLA is a conviction that a sound understanding of fundamental theory is of the utmost

importance. This is developed through a combination of formal instruction and practical computing work, structured in a sequence of courses progressing from introductory to advanced. It is considered essential to continuously make explicit the architectural implications of computing concepts.

REFERENCES:

1. Applied Research of Cambridge, "OXSYS-EDS User's Manual", Cambridge, 1977.

2. Evans and Sutherland Computer Corporation, "The E & S Design System: A Brief Preliminary Description", Mountain View California, 1976.

3. C. Eastman, "General Purpose Building Description Systems", Computer Aided Design, vol. 8, no. 1 (January 1976).

4. C. Eastman and M. Henrion, "Language for a Design Information System", Institute of Physical Planning, Carnegie-Mellon University, Pittsburgh Pennsylvania, 1976.

5. W.J. Mitchell and M. Oliverson, "Criteria for Evaluation of Three-Dimensional Building Description Systems", Applied Research of Cambridge (Canada), Toronto Ontario, 1977.

6. H.A. Simon, The Sciences of the Artificial, MIT Press, 1969.

APPENDIX: LIST OF TEXTS FOR USE IN TEACHING COMPUTER AIDED ARCHITECTURAL DESIGN

General

C.M. Eastman (ed.) Spatial Synthesis in Computer-Aided Building Design, Applied Science, 1975.

J. Gero (ed.) Computer Applications in Architecture, Applied Science (in press).

W.J. Mitchell, Computer Aided Architectural Design, Petrocelli/Charter 1977.

Design theory and method

H.A. Simon, The Sciences of the Artificial, MIT Press, 1969.

W.A. Wickelgren, How to Solve Problems, W.H. Freeman, 1974.

Mathematical modeling

L. March (ed.) The Architecture of Form, Cambridge University Press, 1976.

L. March and P. Steadman, The Geometry of Environment, MIT Press, 1974.

L. Martin and L. March (eds.), Urban Space and Structures, Cambridge University Press, 1972.

Computer Graphics

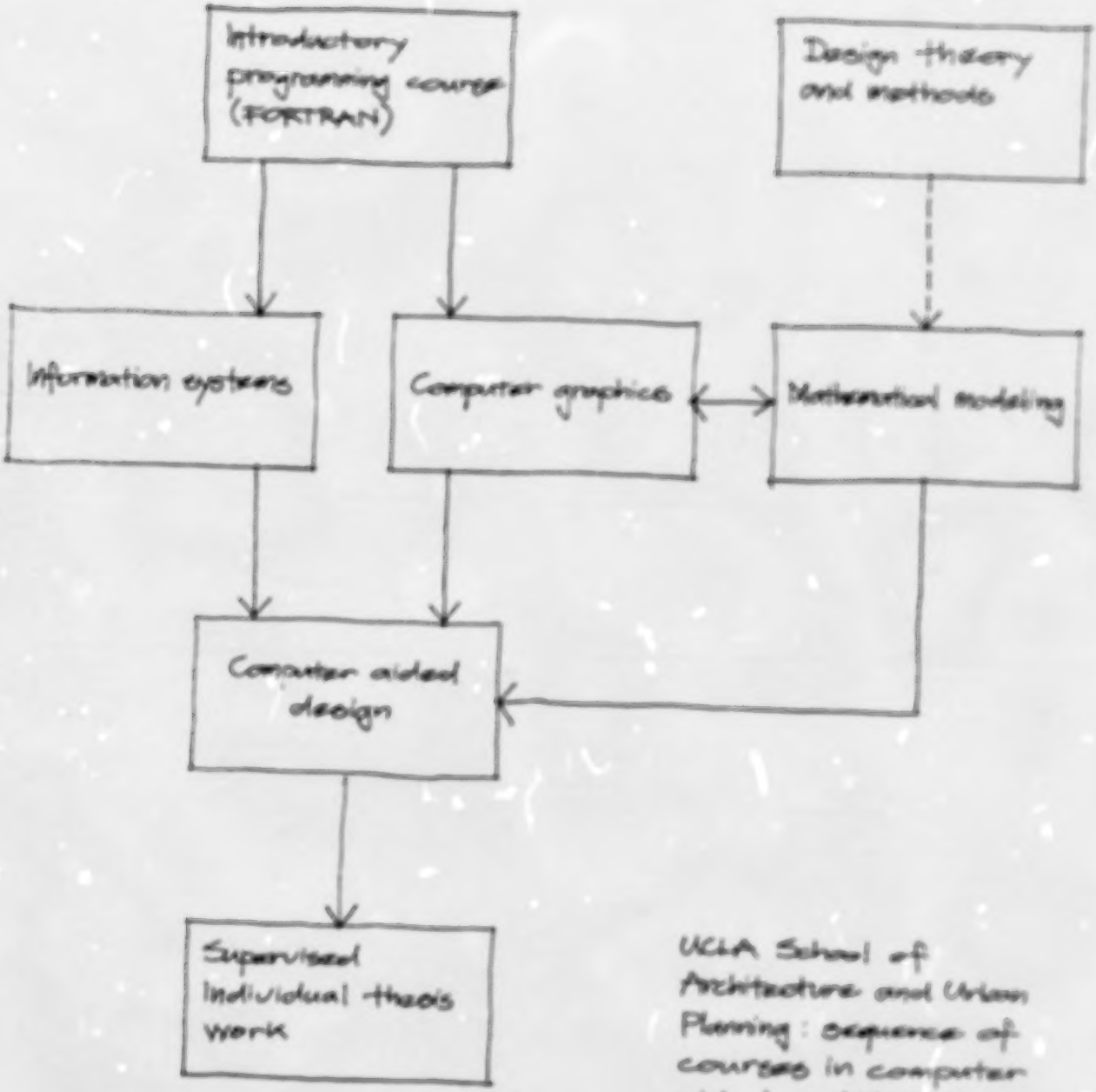
W.M. Newman and R.F. Sproull, Principles of Interactive Computer Graphics, McGraw-Hill, 1973.

D.F. Rogers and J.A. Adams, Mathematical Elements for Computer Graphics, McGraw-Hill, 1976.

Data Base:

C.J. Date, An Introduction to Database Systems, Addison-Wesley, 1976.

J. Martin, Computer Data-Base Organization, Prentice-Hall, 1975.



UCLA School of Architecture and Urban Planning: sequence of courses in computer aided architectural design